Claims:

What is claimed is:

1. A method of performing a Cyclic Redundancy Check (CRC) calculation, said CRC calculation done with N-bit at a time [500] over a binary string of data bits [520], said CRC calculation based on a generator polynomial G(X) [130] of degree d [131], said CRC calculation having intermediate and final results fitting a d-bit wide Field Check Sequence (FCS) [120], said generator polynomial allowing to form a multiplicative cyclic group comprised of d-bit wide binary vectors [400], said method comprising the steps of:

picking [1100] a new N-bit chunk of data bits from said binary string of data bits;

dividing [1110], modulo said generator polynomial G(X), said new N-bit chunk of data bits thus, getting a d-bit wide division result [535];

displacing [1120], in said multiplicative cyclic group, a current value of said d-bit wide FCS, considered as one of said d-bit wide binary vectors, of a value corresponding to said N-bit at a time;

adding [1130], modulo two, said d-bit wide division result and said displaced d-bit wide FCS;

updating [1140] said d-bit wide FCS;

checking if more data bits of said binary string of data bits are left for calculation:

if yes [1151], resuming calculation loop at picking step;

if not [1152], exiting calculation loop;

thereby, getting a final result of said CRC calculation in said d-bit wide FCS.

2. The method according to claim 1 wherein said final result is utilized to generate said d-bit wide FCS [510] for said binary string of data bits.

3. The method according to claims 1 or 2 wherein said final result is a checking result of said binary string of data bits [520] including said d-bit wide FCS [510].

4. The method according to claim 1 wherein said dividing step is replaced, if value of said N-bit is equal to said degree d, by the step of:

handling directly [630] said new N-bit chunk of data bits as if it is said d-bit wide division result [535].

5. The method according to claim 4 wherein said handling step includes, if value of said N-bit is lower than said degree d, the further step of:

padding said new N-bit chunk of data with enough leading zeros to match said d-bit wide FCS [540].

6. The method according to claim 1 wherein said CRC calculation is done from a most significant bit (MSB) [530] of said binary string of data bits and wherein said displacing step includes a forward multiplication [560] of said d-bit wide FCS.

7. The method according to claim 1 wherein said CRC calculation is done from a least significant bit (LSB) [710] of said binary string of data bits and wherein said displacing step includes a backward multiplication [760] of said d-bit wide FCS.

8. The method according to claim 1 wherein said binary string of data bits is a frame or message moved over a communications network.

9. The method according to claim 1 wherein said binary string of data bits is derived or stored as a file in a computing system.

10. A method for generating a combinational logic block [1030] implementing a Divider Modulo G(X) [535] and a xN Multiplier [560], said Divider Modulo G(X) and said xN Multiplier to permit a forward calculation be done said N-bit at a time on the basis of said generator polynomial G(X) of degree d, said method comprising the steps of:

starting [1200] from an identity vector (α^0) [420] part of said multiplicative group [400] formed of said d-bit wide binary vectors with said generator polynomial G(X) of degree d;

recording [1210] said d-bit wide binary vector;

checking [1230] if at least N+d of said d-bit wide binary vectors have been generated;

if not:

multiplying [1220] a last recorded of said d-bit wide binary vector by a second d-bit wide binary vector (α^1) [430] part of said multiplicative group [400];

resuming at said recording step [1210];

if yes [1232]:

creating [1240] said Divider Modulo G(X) and said xN Multiplier out of said at least N+d d-bit wide binary vectors;

thereby, obtaining said combinational logic block.

11. A method for generating a combinational logic block implementing a Divider Modulo G(X) and a $x^{-}N$ Multiplier [760], said Divider Modulo G(X) and said $x^{-}N$ Multiplier to permit that backward calculations be done said N-bit at a time on the basis of said generator polynomial G(X) of degree d, said method comprising the steps of:

starting [1300] from said identity vector (α^0) [420] part of said multiplicative group [400] formed of said d-bit wide binary vectors with said generator polynomial G(X) of degree d;

recording [1310] said d-bit wide binary vector;

checking [1330] if at least N of said d-bit wide binary vectors have been generated;

if not:

multiplying [1320] said last recorded of said d-bit wide binary vector by said second d-bit wide binary vector (α^1) [430] part of said multiplicative group [400];

resuming at said recording step [1310];

if yes [1332]:

deducing [1340] a last d-bit wide binary vector (α^{-1}) [910] part of said multiplicative group;

restarting [1350] from said identity vector (α^0) [420];

multiplying [1360] by said last d-bit wide binary vector (α^{-1});

keep recording [1370] said d-bit wide binary vector;

checking [1380] if at least N new of said d-bit wide binary vectors have been generated;

if not:

resuming at said multiplying by said last d-bit wide binary vector (α^{-1}) step [1360];

if yes [1382]:

creating [1390] said Divider Modulo G(X) and said x^{-N} Multiplier out of said at least N d-bit wide binary vectors and said at least N new d-bit wide binary vectors;

thereby, obtaining said combinational logic block.

12. A system, in particular a processor [1400], executing instructions for carrying out CRC calculations according to the method of any one of the claims 1 to 9.

13. A system, in particular a state machine [1000] aimed at performing CRC calculations N-bit at a time, comprising means adapted for carrying out the method according to any one of the claims 1 to 9.

14. A system, in particular a work station [1500], comprising means adapted for generating a logic block according to the methods of claims 10 or 11.

15. A computer-like readable medium comprising instructions for carrying out the methods according to any one of the claims 1 to 11.

16. A method for calculating Cyclic Redundancy Check (CRC) including the acts of:

(a) selecting N-bits, N greater than 1, of data from a binary string of data bits;

(b) displaying (1120) in a multiplicative cyclic group of values corresponding to N a current value of d-bit wide FCS, considered as one of the d-bit wide binary vectors;

(c) adding (1130) modulo two, the N-bits and said displaced d-bit wide FCS; and

(d) updating said d-bit wide FCS.

17. The method of claim 16 further including the acts of:

(e) checking if more bits of said binary string are left for calculation;

(f) if yes, repeating acts (a) through (e);

(g) if not, existing with result in act (d) being the calculated CRC.

18. The method of claims 16 or 17 further including the step of prior to performing step (b) dividing the N-bits, modulo generator polynomial G(x), to obtain a d-bit wide division result.